

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ  
Національний аерокосмічний університет ім. М. Є. Жуковського  
«Харківський авіаційний інститут»

Факультет систем управління літальних апаратів  
Кафедра систем управління літальних апаратів

## **Лабораторна робота № 10**

з дисципліни «Алгоритмізація та програмування»  
на тему «"Створення і обробка структур даних мовою C ++

XAI.301.312.2

Виконав студент гр. 312

Ксенія ВЕЛІКОДАНОВА  
(підпис, дата) (П.І.Б.)

Перевірив

к.т.н., доц. Олена ГАВРИЛЕНКО  
(підпис, дата) (П.І.Б.)

## МЕТА РОБОТИ

Вивчити теоретичний матеріал з основ представлення структур (записів) мовою C ++, а також їх передачі в функції, і реалізувати декларування і обробку структур мовою C ++ в середовищі Visual Studio.

## ПОСТАНОВКА ЗАДАЧІ

### Завдання 1.

Вирішити задачу зі структурами даних. Param76 з табл.1.

### Завдання 2.

Для задач Begin18 та Boolean22. з табл.2-3:

- A. Описати структуру, яка містить всі вхідні і всі вихідні дані задачі.
- B. Визначити функцію, що реалізує обробку структури відповідно до задачі.
- C. Визначити функцію, що перевіряє на коректність і заповнює відповідні поля вхідних даних структури
- D. Викликати функції з пунктів C, B після оголошення змінної (об'єкту) структури.
- E. Вивести значення полів вихідних даних.

### Завдання 3.

Рішення всіх трьох задач реалізувати в одному консольному додатку, структурувати на модулі.

## ВИКОНАННЯ РОБОТИ

### Завдання 1.

Вирішення задачі Param76

Вхідні дані (ім'я, опис, тип, обмеження):

```
struct TTime {
    int Hour; // 0–23
    int Min;  // 0–59
    int Sec;  // 0–59
};
```

Вихідні дані (ім'я, опис, тип):

TTime& t

Алгоритм вирішення показано на рис. 1

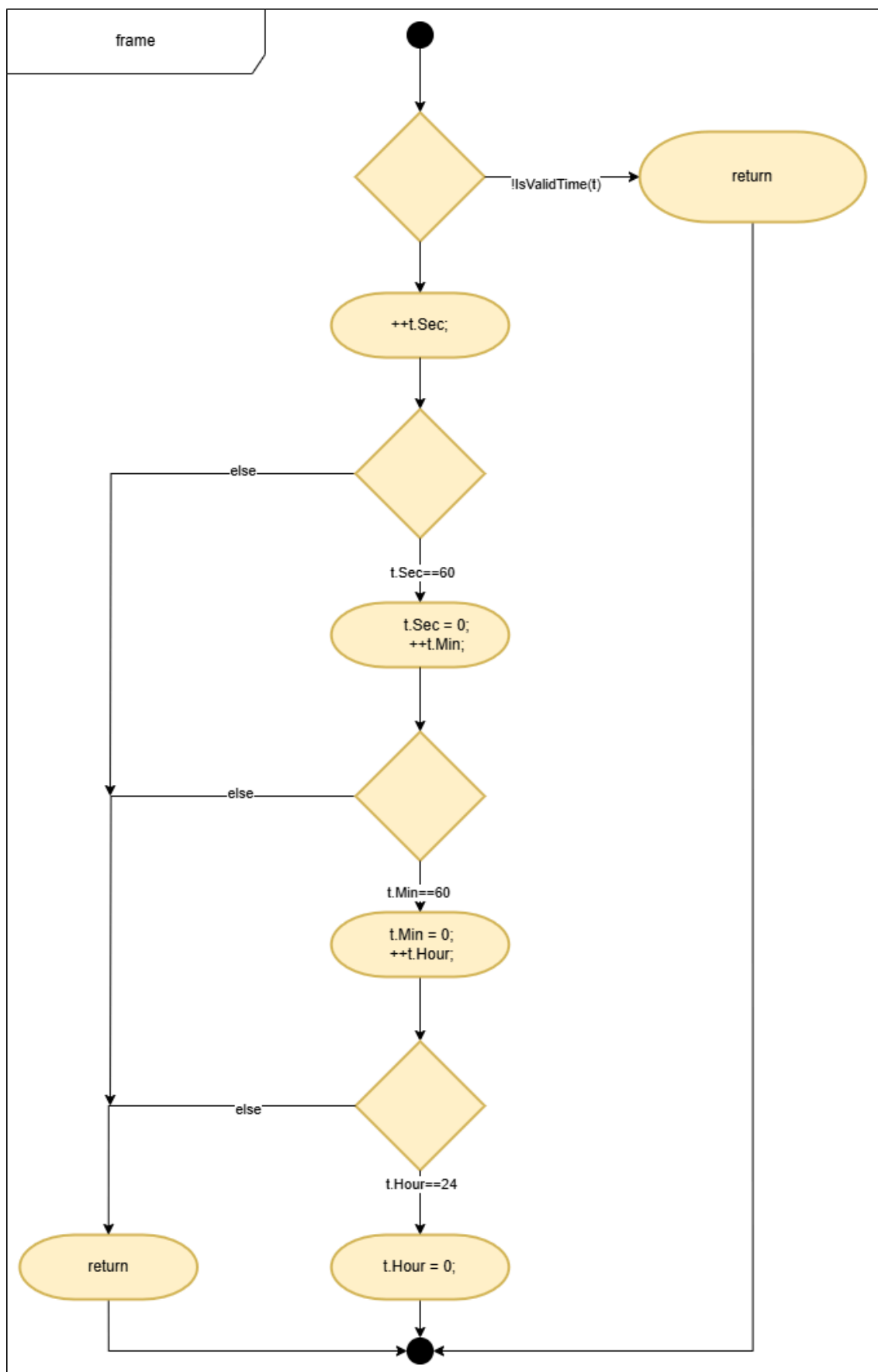


Рисунок 1 – Алгоритм Param76

Лістинг коду вирішення задачі Param76 наведено в дод. А (стор. 6).  
Екран роботи програми показаний на рис. Б.1.

Завдання 2.

Вирішення задачі Begin18

Вхідні дані (ім'я, опис, тип, обмеження):

double A, B, C

Вихідні дані (ім'я, опис, тип):

double AC, BC, sum

Алгоритм вирішення показано на рис. 2

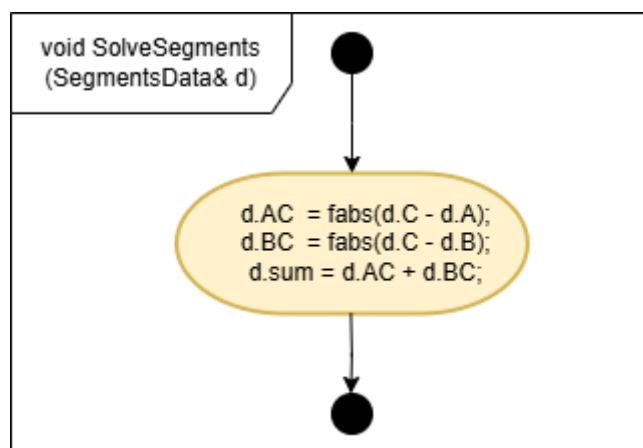


Рисунок 2 – Алгоритм Begin18

Лістинг коду вирішення задачі Begin18 наведено в дод. А (стор. 6).  
Екран роботи програми показаний на рис. Б.2.

Завдання 3.

Вирішення задачі Boolean22

Вхідні дані (ім'я, опис, тип, обмеження):

int num

Вихідні дані (ім'я, опис, тип):

bool monotonic

Алгоритм вирішення показано на рис. 3

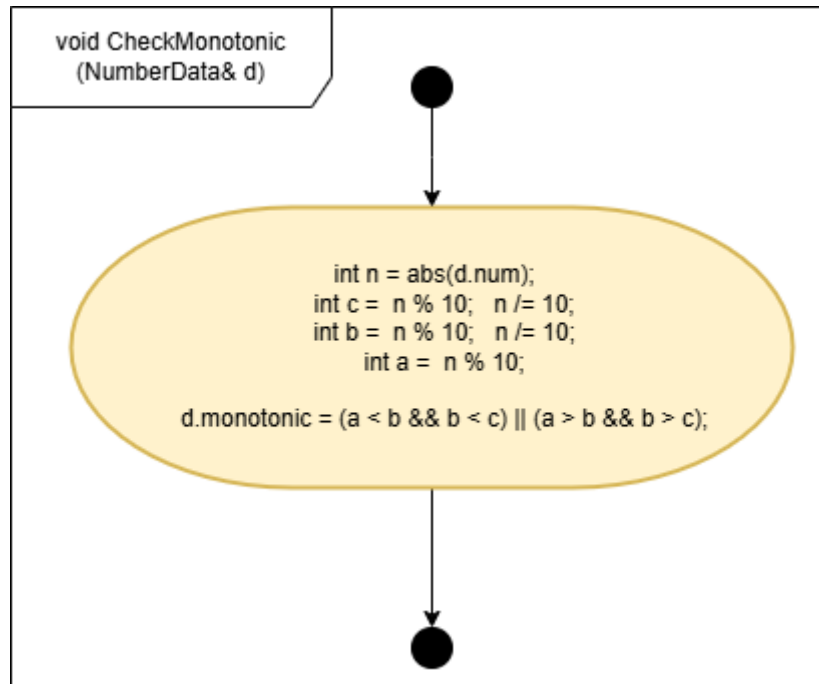


Рисунок 3 – Алгоритм Boolean22

Лістинг коду вирішення задачі Boolean22 наведено в дод. А (стор. 6).  
Екран роботи програми показаний на рис. Б.3.

## ВИСНОВКИ

Було вивчено принципи створення та використання структур даних у мові С++. Закріплено на практиці навички перевірки коректності вхідних даних, реалізації процедур обробки та організації коду у вигляді модулів. Відпрацьовано в коді роботу з часом, координатами та числовими послідовностями.

## ДОДАТОК А

### Лістинг коду програми

```
#include <iostream>
#include <cmath>

using namespace std;

/*=====*/
/*                                TASK 1                                */
/*=====*/
struct TTime {
    int Hour;   // 0-23
    int Min;    // 0-59
    int Sec;    // 0-59
};

/*- перевірка коректності часу -*/
bool IsValidTime (const TTime& t)
{
    return (t.Hour>=0 && t.Hour<24) &&
           (t.Min >=0 && t.Min <60) &&
           (t.Sec >=0 && t.Sec <60);
}

/*- додаємо одну секунду, якщо час коректний -*/
void NextSec (TTime& t)
{
    if (!IsValidTime(t)) return;

    ++t.Sec;
    if (t.Sec==60) {
        t.Sec = 0;
        ++t.Min;
        if (t.Min==60) {
            t.Min = 0;
            ++t.Hour;
            if (t.Hour==24) t.Hour = 0;
        }
    }
}

/*=====*/
/*                                TASK 2  (two subtasks)                                */
/*=====*/

/*----- 2.1  Відрізки на числовій осі -----*/
```

```

struct SegmentsData {
    double A{}, B{}, C{};
    double AC{}, BC{}, sum{};
};

/*- введення та перевірка -*/
void ReadSegmentsData (SegmentsData& d)
{
    cout << "\n[2.1] Введіть три точки A, B, C: ";
    cin >> d.A >> d.B >> d.C;
    // коректність тут формально не обмежується: дійсні числа дозволені
}

/*- обробка -*/
void SolveSegments (SegmentsData& d)
{
    d.AC = fabs(d.C - d.A);
    d.BC = fabs(d.C - d.B);
    d.sum = d.AC + d.BC;
}

/*----- 2.2 Тризначне число: зростання/спадання -----*/
struct NumberData {
    int num{};
    bool monotonic{}; // істинність висловлювання
};

/*- введення з перевіркою на тризначність -*/
void ReadNumberData (NumberData& d)
{
    do {
        cout << "\n[2.2] Введіть тризначне число: ";
        cin >> d.num;
    } while (abs(d.num) < 100 || abs(d.num) > 999);
}

/*- обробка -*/
void CheckMonotonic (NumberData& d)
{
    int n = abs(d.num);
    int c = n % 10;    n /= 10;
    int b = n % 10;    n /= 10;
    int a = n % 10;

    d.monotonic = (a < b && b < c) || (a > b && b > c);
}

/*=====*/
/*                                MAIN                                */
/*=====*/

```

```

int main()
{
    /*----- TASK 1 tests -----*/
    cout << "===== TASK 1: NextSec =====\n";
    for (int i=1;i<=5;++i)
    {
        TTime t;
        cout << "Час #" << i << " (год хв сек): ";
        cin >> t.Hour >> t.Min >> t.Sec;

        cout << " Вхідні дані: ";
        if (IsValidTime(t))
            cout << t.Hour << ':' << t.Min << ':' << t.Sec << '\n';
        else
            cout << "(некоректний час)\n";

        NextSec(t);

        cout << " Після NextSec: ";
        if (IsValidTime(t))
            cout << t.Hour << ':' << t.Min << ':' << t.Sec << "\n\n";
        else
            cout << "(час не змінено, був некоректний)\n\n";
    }

    /*----- TASK 2.1 -----*/
    SegmentsData seg;
    ReadSegmentsData(seg);
    SolveSegments(seg);
    cout << "\n===== TASK 2.1: Segments =====\n";
    cout << "AC = " << seg.AC << "\n"
        << "BC = " << seg.BC << "\n"
        << "AC+BC = " << seg.sum << "\n";

    /*----- TASK 2.2 -----*/
    NumberData nd;
    ReadNumberData(nd);
    CheckMonotonic(nd);
    cout << "\n===== TASK 2.2: Number =====\n";
    cout << "Цифри числа " << nd.num
        << (nd.monotonic ? " дійсно " : " не ")
        << "утворюють зростаючу або спадаючу послідовність.\n";

    cout << "\n*** Програма завершила роботу ***\n";
    return 0;
}

```



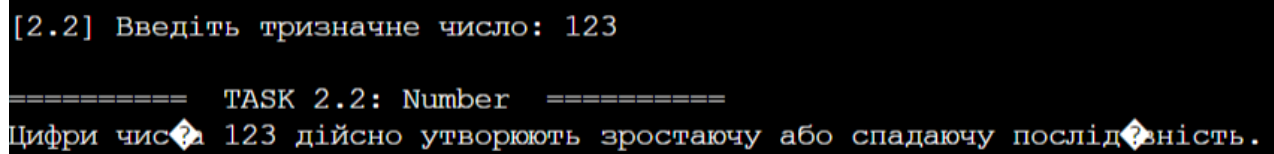
ДОДАТОК Б  
Скрін-шоти вікна виконання програми

```
===== TASK 1: NextSec =====  
Час #1 (год хв сек): 1 1 1  
Вхідні дані: 1:1:1  
Після NextSec: 1:1:2  
  
Час #2 (год хв сек): 25 1 1  
Вхідні дані: (некоректний час)  
Після NextSec: (час не змінено, був некоректний)
```

Рисунок Б.1 – Екран виконання програми для вирішення завдання  
Param76

```
[2.1] Введіть три точки A, B, C: 1 3 5  
  
===== TASK 2.1: Segments =====  
AC = 4  
BC = 2  
AC+BC = 6
```

Рисунок Б.2 – Екран виконання програми для вирішення завдання  
Begin18



[2.2] Введіть тризначне число: 123

===== TASK 2.2: Number =====

Цифри числа 123 дійсно утворюють зростаючу або спадаючу послідовність.

Рисунок Б.3 – Екран виконання програми для вирішення завдання  
Boolean22